

Direct Manipulation of Interactive Character Skins



Alex Mohr

Luke Tokheim

Michael Gleicher

University of
Wisconsin, Madison



Introduction

- Characters for interactive systems
 - Speed is king
- Linear Blend Skinning (SSD, etc.)
 - Widely used method
- Pros: fast, hardware accelerated
- Cons: hard to author, can look bad
- We address the former



Example

- Current systems
 - *Indirect* manipulation interface
 - Paint parameters over meshes
 - Unclear what is possible
- Instead, let users edit *directly*
 - Drag points explicitly
 - Compute closest achievable position
 - Solve for skin parameters

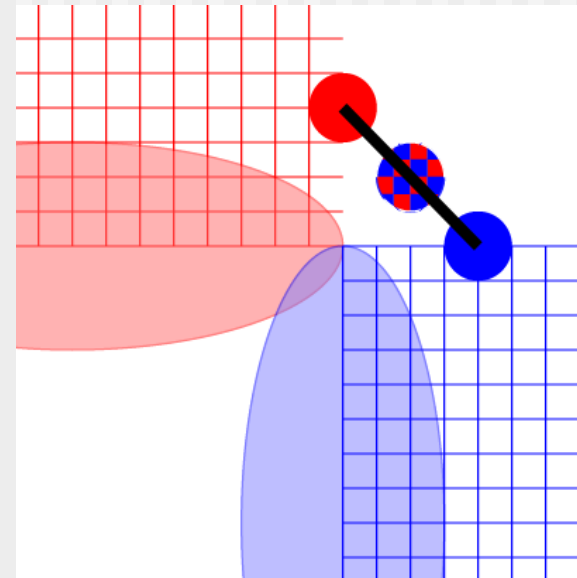
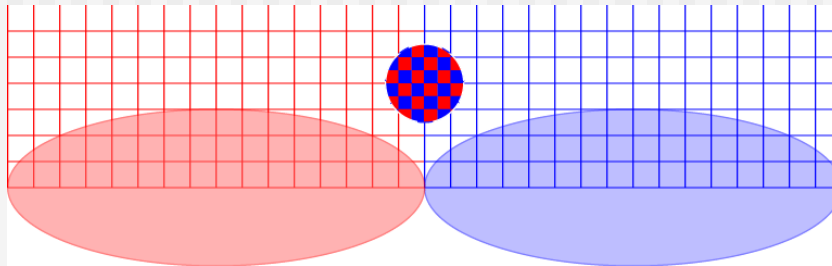


Video



Linear Blend Skinning

- [Catmull '72] [Magenat-Thalmann et al. '88] [Lewis et al. '00]
- Place skeleton inside geometry (dress pose)
- Joint ' Transformation matrix
- Linear blend of joint matrices transforms points





Linear Blend Skin Computation

- Vertex deformed by linear blend of matrices
- Weights affine, usually convex

$$\bar{\mathbf{v}} = \sum_{i=1}^n w_i R_i \mathbf{v}_d$$

- R_i = transformation matrix associated with i th joint



Linear Blend Skin Authoring

- Must determine

- Influence set
- Vertex weights

- Current systems use “painting” UI

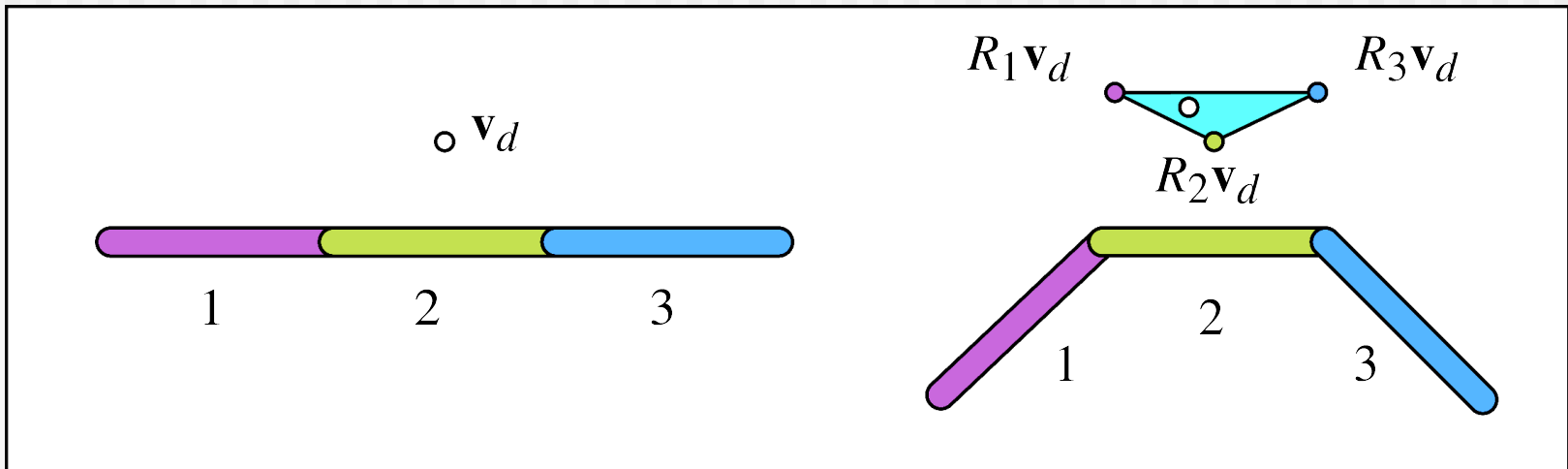
- Select joint, weight, then draw on mesh
- What deformations are possible?

$$\bar{\mathbf{v}} = \sum_{i=1}^n w_i R_i \mathbf{v}_d$$

Range of Possible Deformations

- Valid subspace: What is reachable?
- Affine or convex hull of rigidly transformed vertex

$$\bar{\mathbf{v}} = \sum_{i=1}^n w_i R_i \mathbf{v}_d$$

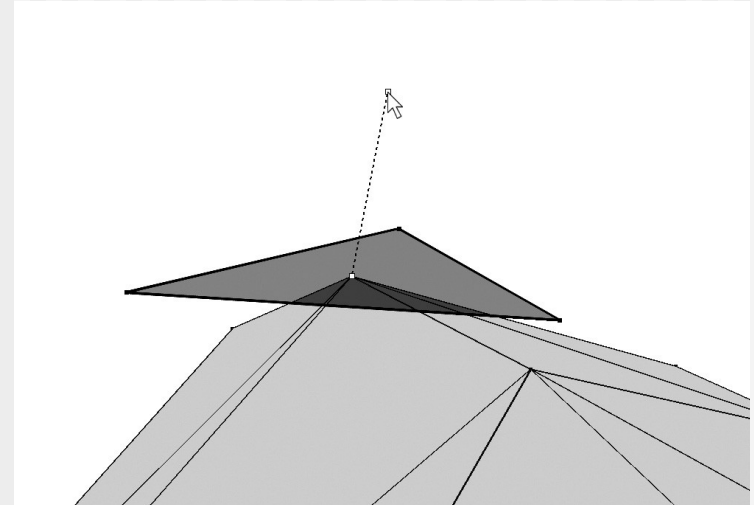


Direct Manipulation

- Algorithm
 - Project onto valid subspace (target \mathbf{t})
 - Determine weights so skin achieves \mathbf{t}

Direct manip. is well known
[Hutchins et al. '86]
[Schneiderman '83]
[Sutherland '63]

Similar work
[Fowler '92]
[Hsu et al. '92]





Computing Weights

Let: $w_1 = 1 - \sum_{i=2}^n w_i$

Solve:

$$\left[(R_2 - R_1) \mathbf{v}_d \cdots (R_n - R_1) \mathbf{v}_d \right] \begin{bmatrix} w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} = [\mathbf{t} - R_1 \mathbf{v}_d]$$

Guarantees weights are affine.
Does not guarantee weights
are convex.



Video





Degeneracies

- If some columns linearly dependent

$$\left[(R_2 - R_1)\mathbf{v}_d \cdots (R_n - R_1)\mathbf{v}_d \right] \begin{bmatrix} w_2 \\ w_3 \\ \vdots \\ w_n \end{bmatrix} = [\mathbf{t} - R_1\mathbf{v}_d]$$

- Practically
 - More than four influences
 - Valid subspace collapsed in some way
- Happens frequently in practice
 - More than four influences desirable
 - User moves just one joint



Handling Degeneracies

- Quick fix?
 - Disallow more than four influences
 - Perturb joints in other cases
- Geometric projection *guarantees* a convex solution
- Solve a linear program to enforce convexity constraints



Linear Program

- Objective: minimize L_1 change in weights

$$\text{Constraint: } -\mathbf{y} \leq \mathbf{w} - \mathbf{w}_p \leq \mathbf{y}$$

$$\text{Objective: } \min \sum y_i$$

- Constraints

$$\begin{bmatrix} R_1 \mathbf{v}_d \cdots R_n \mathbf{v}_d \end{bmatrix} \mathbf{w} = \mathbf{t}$$

$$\sum_{i=1}^n w_i = 1$$

$$\forall i, w_i \geq 0$$

This always has a feasible solution due to the projection step.



Discussion

- Allow users to move groups of vertices directly
- Skin parameters computed automatically
- Allow users to see possible range of deformations
- View edits in multiple poses



Video





Summary

- Method for improving skin authoring for interactive characters
- Users must still define influence sets
- Much more intuitive interaction
 - As close as possible to user's wishes
 - Clear what is possible
- Compatible with current interface



Thanks

- UW Graphics Group
- NSF Grant CCR-9984506
- NSF Grant CCR-0204372
- Intel